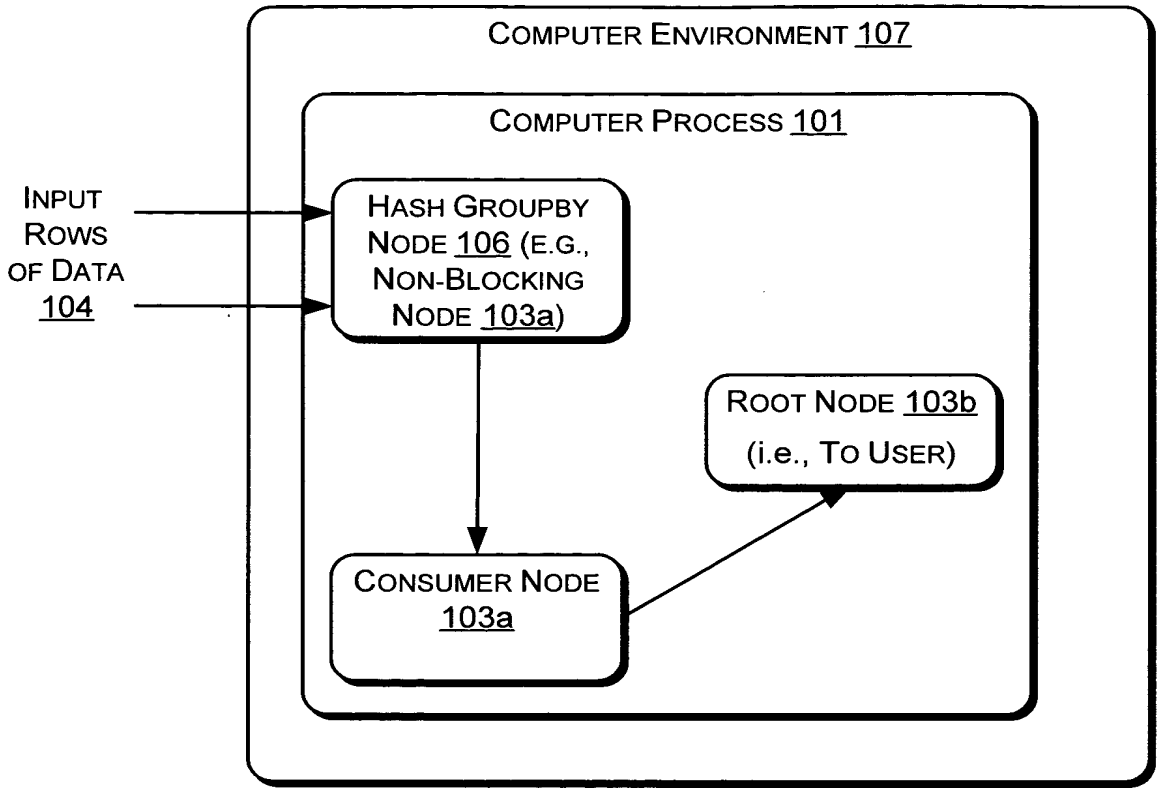
COMPUTER ENVIRONMENT 107
COMPUTER PROCESS 101
INPUT ROWS OF DATA 104
HASH GROUPBY NODE 106 (E.G., NON-BLOCKING NODE 103a)
ROOT NODE 103b (i.e., TO USER)
CONSUMER NODE 103a

Fig. 1

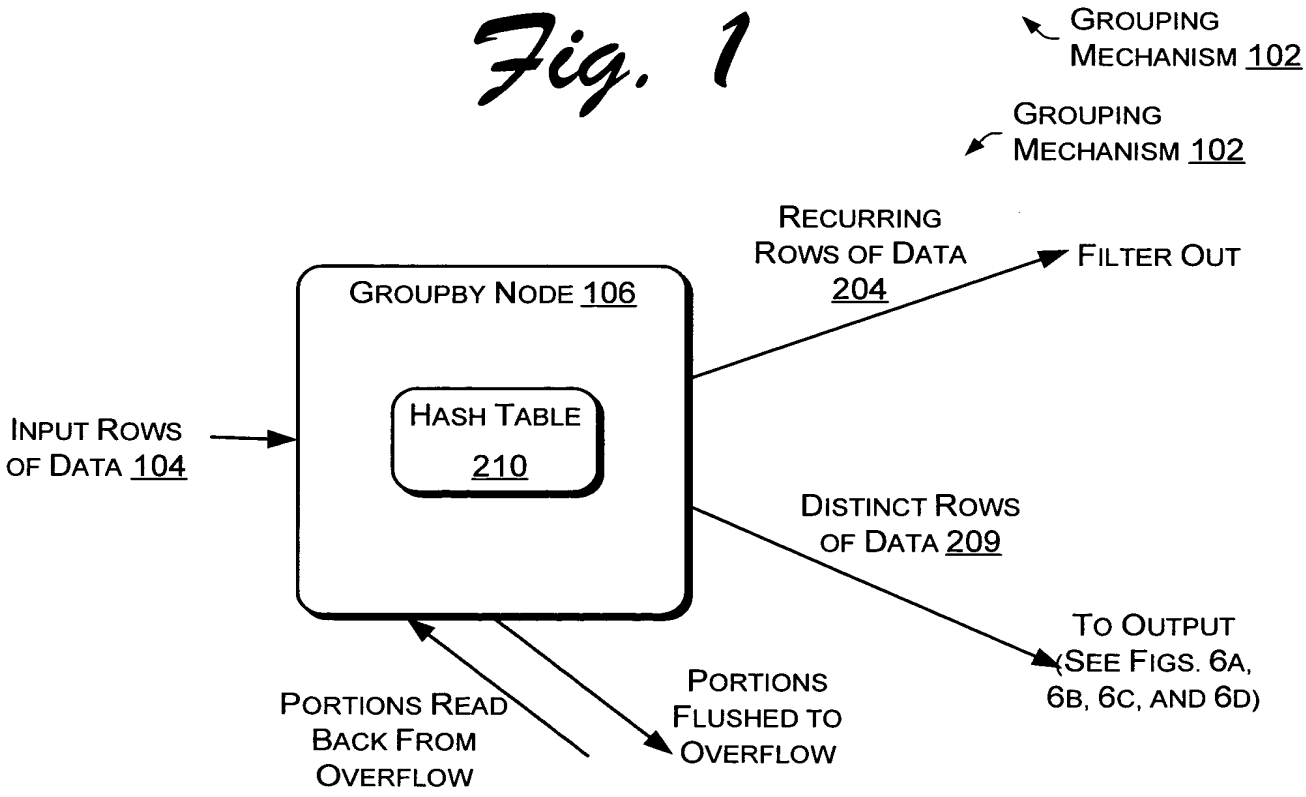
GROUPING MECHANISM 102
GROUPING MECHANISM 102
RECURRING ROWS OF DATA 204
FILTER OUT
GROUPBY NODE 106
HASH TABLE 210
INPUT ROWS OF DATA 104
DISTINCT ROWS OF DATA 209
TO OUTPUT (SEE FIGS. 6A, 6B, 6C, AND 6D)
PORTIONS READ BACK FROM OVERFLOW
PORTIONS FLUSHED TO OVERFLOW

Fig. 2

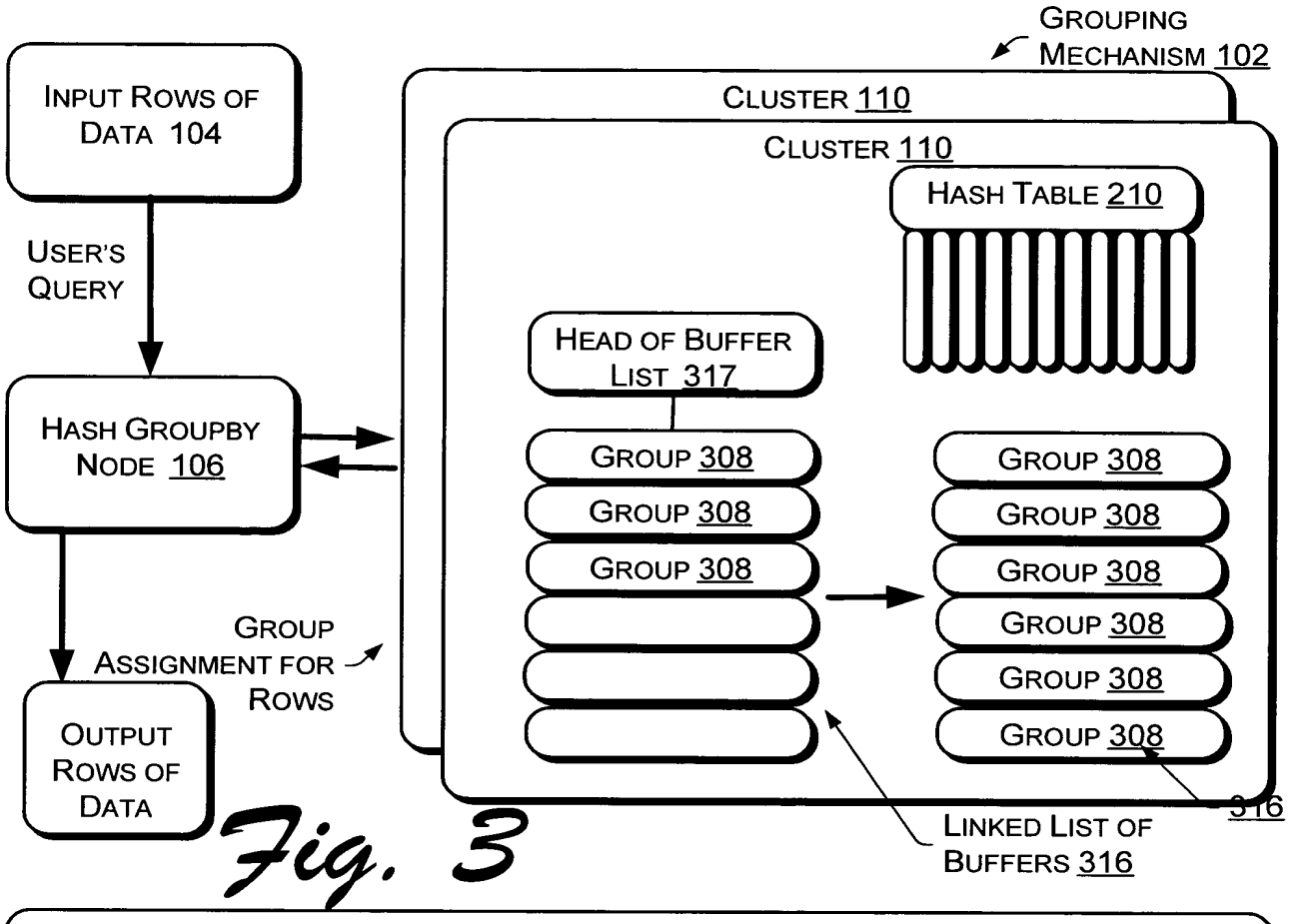
GROUPING MECHANISM 102
INPUT ROWS OF DATA 104
USER'S QUERY
HASH GROUPBY NODE 106
GROUP ASSIGNMENT FOR ROWS
OUTPUT ROWS OF DATA
CLUSTER 110
CLUSTER 110
HASH TABLE 210
HEAD OF BUFFER LIST 317
GROUP 308
GROUP 308
GROUP 308
GROUP 308
GROUP 308
GROUP 308
GROUP 308
GROUP 308
GROUP 308
316
LINKED LIST OF BUFFERS 316

Fig. 3

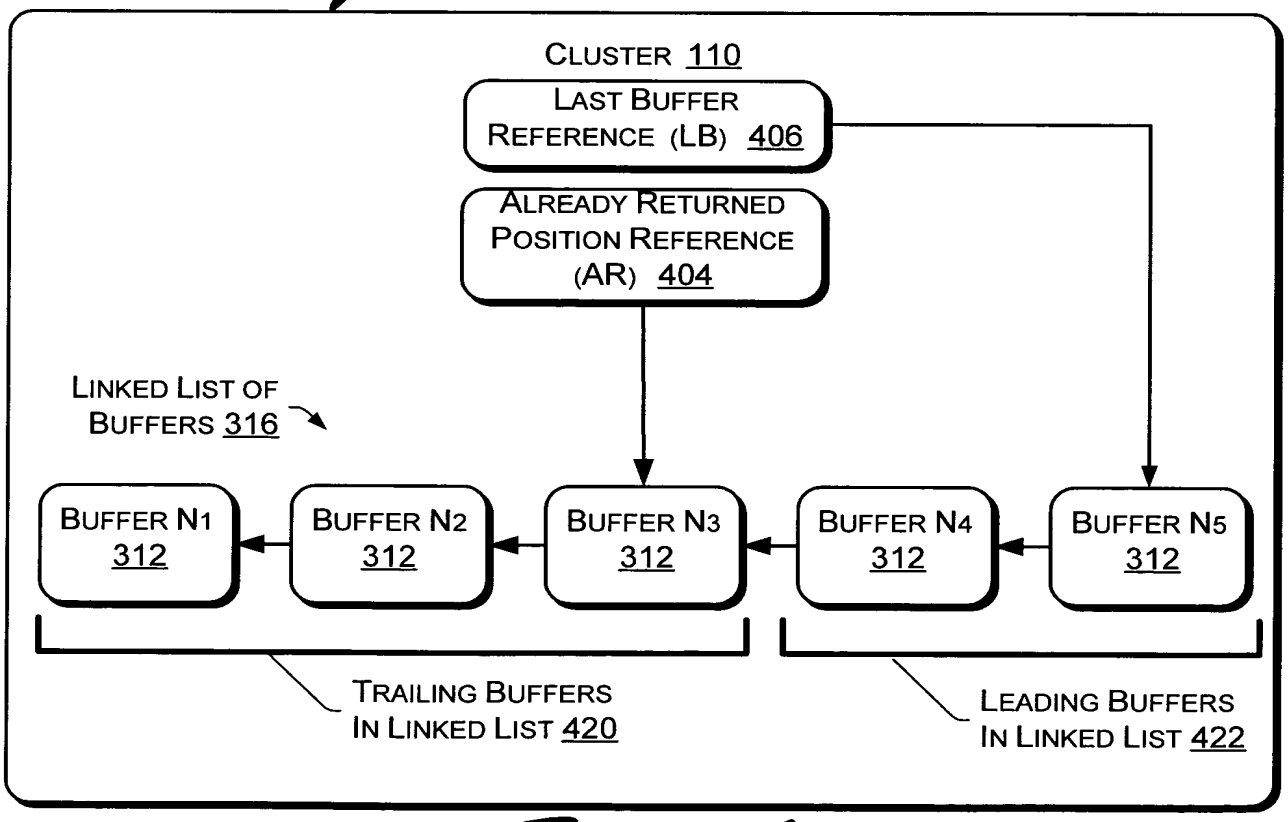
CLUSTER 110
LAST BUFFER REFERENCE (LB) 406
ALREADY RETURNED POSITION REFERENCE (AR) 404
LINKED LIST OF BUFFERS 316
BUFFER N1 312
BUFFER N2 312
BUFFER N3 312
BUFFER N4 312
BUFFER N5 312
TRAILING BUFFERS IN LINKED LIST 420
LEADING BUFFERS IN LINKED LIST 422

Fig. 4

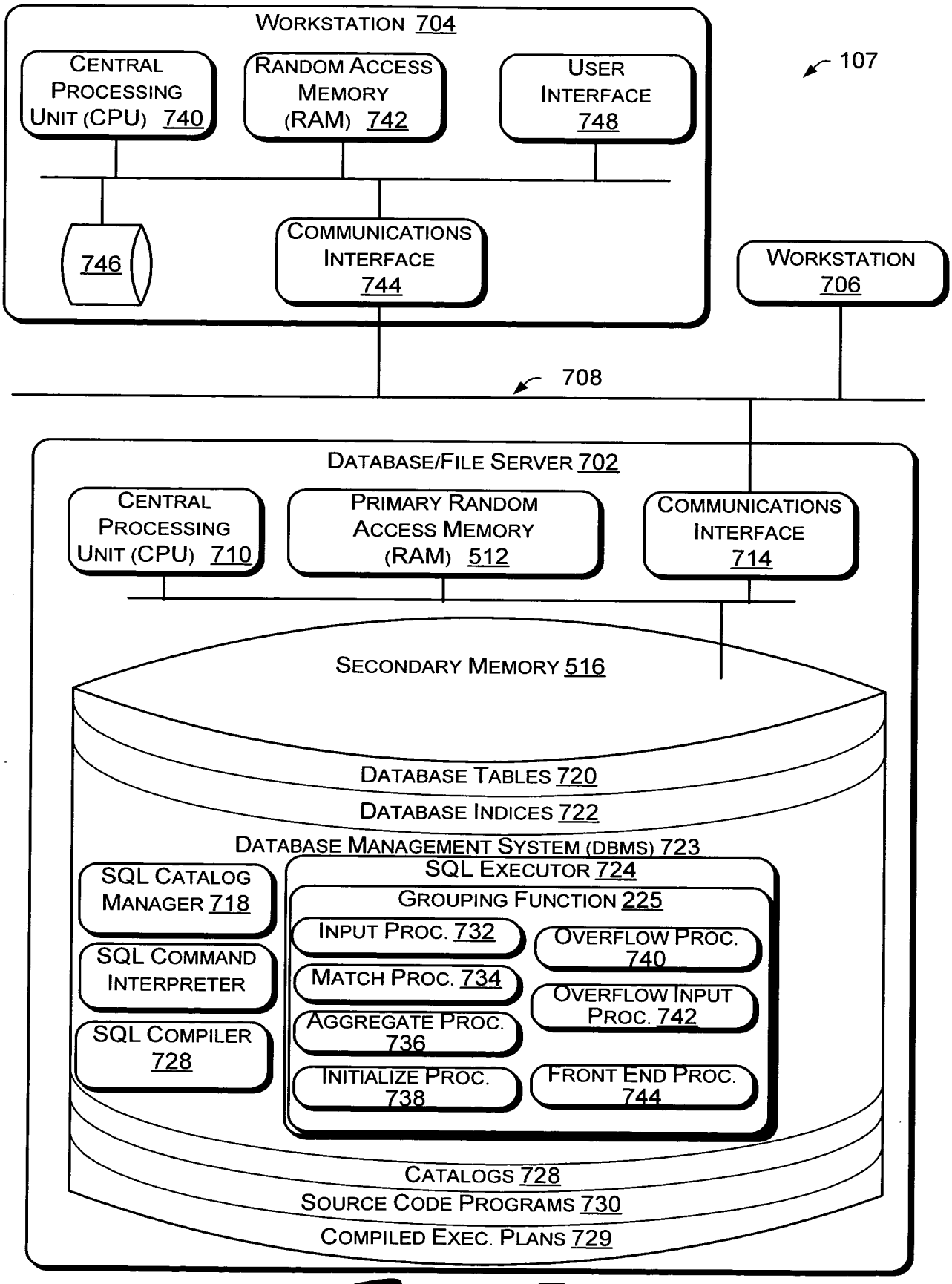
WORKSTATION 704
CENTRAL PROCESSING UNIT (CPU) 740
RANDOM ACCESS MEMORY (RAM) 742
USER INTERFACE 748
107
746
COMMUNICATIONS INTERFACE 744
WORKSTATION 706
708
DATABASE/FILE SERVER 702
CENTRAL PROCESSING UNIT (CPU) 710
PRIMARY RANDOM ACCESS MEMORY (RAM) 512
COMMUNICATIONS INTERFACE 714
SECONDARY MEMORY 516
DATABASE TABLES 720
DATABASE INDICES 722
DATABASE MANAGEMENT SYSTEM (DBMS) 723
SQL CATALOG MANAGER 718
SQL COMMAND INTERPRETER
SQL COMPILER 728
SQL EXECUTOR 724
GROUPING FUNCTION 225
INPUT PROC. 732
MATCH PROC. 734
AGGREGATE PROC. 736
INITIALIZE PROC. 738
OVERFLOW PROC. 740
OVERFLOW INPUT PROC. 742
FRONT END PROC. 744
CATALOGS 728
SOURCE CODE PROGRAMS 730
COMPILED EXEC. PLANS 729

Fig. 5

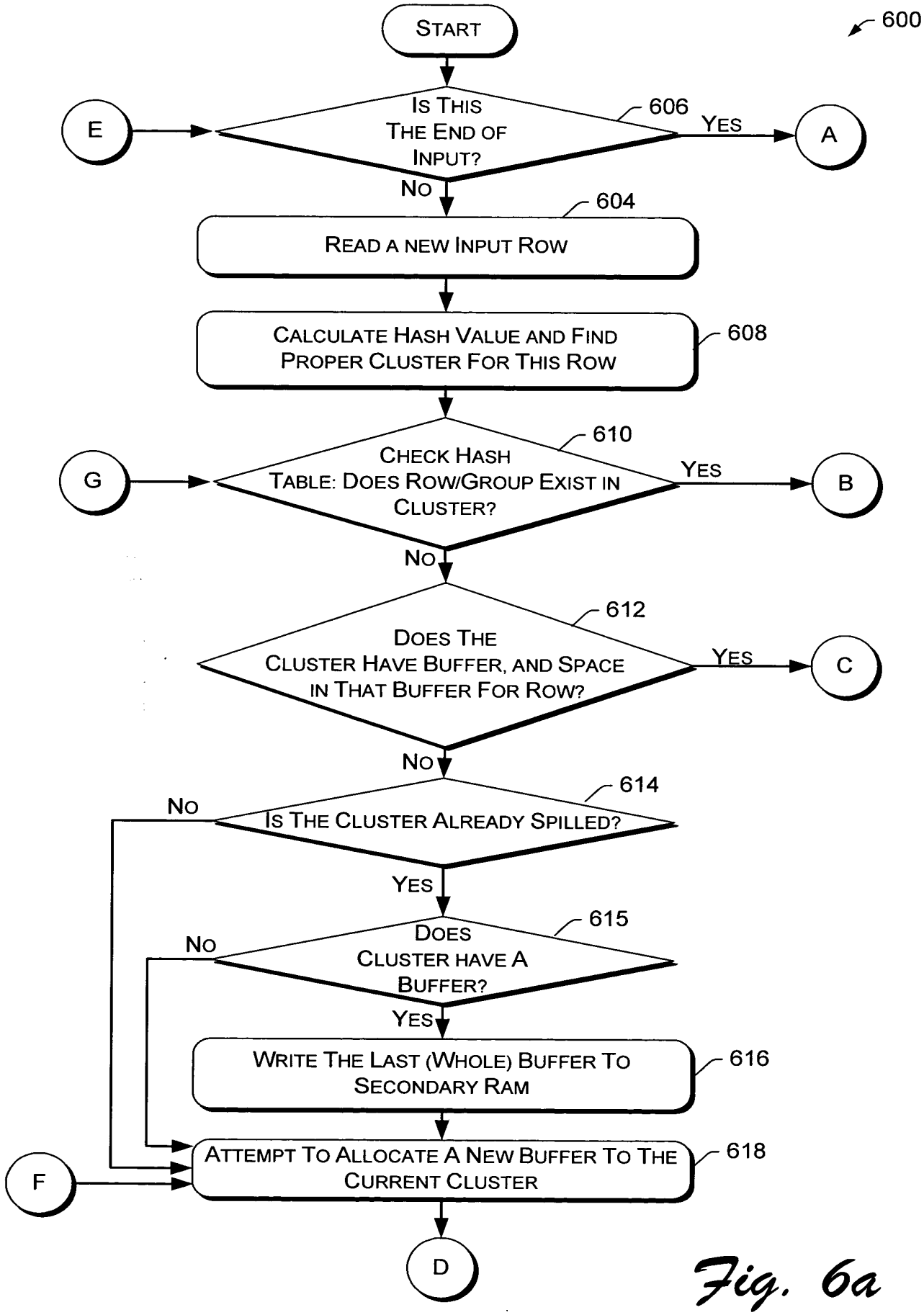
600
START
IS THIS THE END OF INPUT?
606
YES
A
E
NO
604
READ A NEW INPUT ROW
CALCULATE HASH VALUE AND FIND PROPER CLUSTER FOR THIS ROW
608
CHECK HASH TABLE: DOES ROW/GROUP EXIST IN CLUSTER?
610
YES
B
G
NO
DOES THE CLUSTER HAVE BUFFER, AND SPACE IN THAT BUFFER FOR ROW?
612
YES
C
NO
IS THE CLUSTER ALREADY SPILLED?
614
NO
YES
DOES CLUSTER HAVE A BUFFER?
615
NO
YES
WRITE THE LAST (WHOLE) BUFFER TO SECONDARY RAM
616
ATTEMPT TO ALLOCATE A NEW BUFFER TO THE CURRENT CLUSTER
618
F
D

Fig. 6a

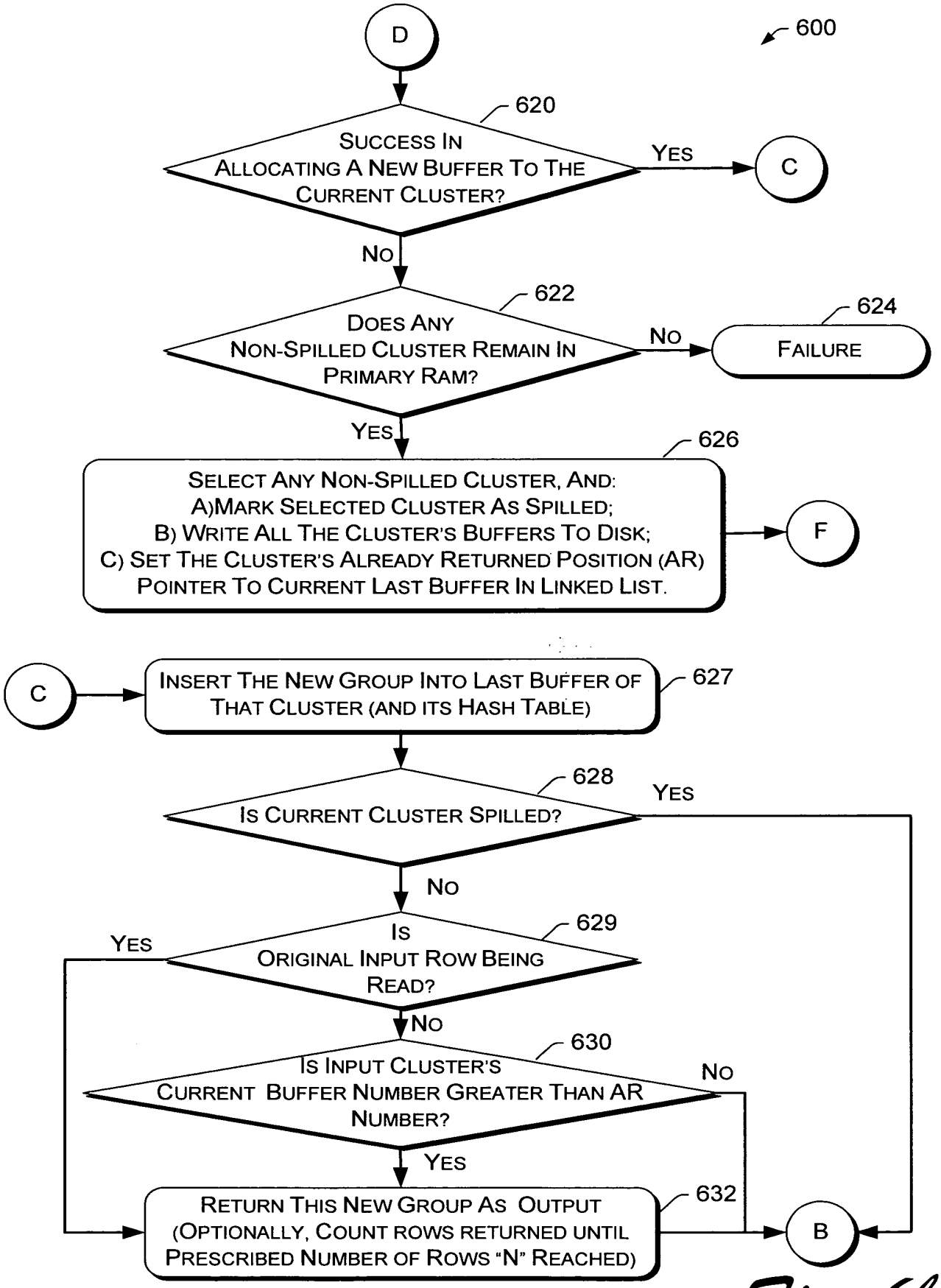
600
D
620
SUCCESS IN ALLOCATING A NEW BUFFER TO THE CURRENT CLUSTER?
YES
C
NO
622
DOES ANY NON-SPILLED CLUSTER REMAIN IN PRIMARY RAM?
NO
624
FAILURE
YES
626
SELECT ANY NON-SPILLED CLUSTER, AND:
A)MARK SELECTED CLUSTER AS SPILLED;
B) WRITE ALL THE CLUSTER'S BUFFERS TO DISK;
C) SET THE CLUSTER'S ALREADY RETURNED POSITION (AR) POINTER TO CURRENT LAST BUFFER IN LINKED LIST.
F
C
INSERT THE NEW GROUP INTO LAST BUFFER OF THAT CLUSTER (AND ITS HASH TABLE)
627
628
IS CURRENT CLUSTER SPILLED?
YES
NO
629
IS ORIGINAL INPUT ROW BEING READ?
YES
NO
630
IS INPUT CLUSTER'S CURRENT BUFFER NUMBER GREATER THAN AR NUMBER?
NO
YES
632
RETURN THIS NEW GROUP AS OUTPUT (OPTIONALLY, COUNT ROWS RETURNED UNTIL PRESCRIBED NUMBER OF ROWS "N" REACHED)
B

Fig. 6b

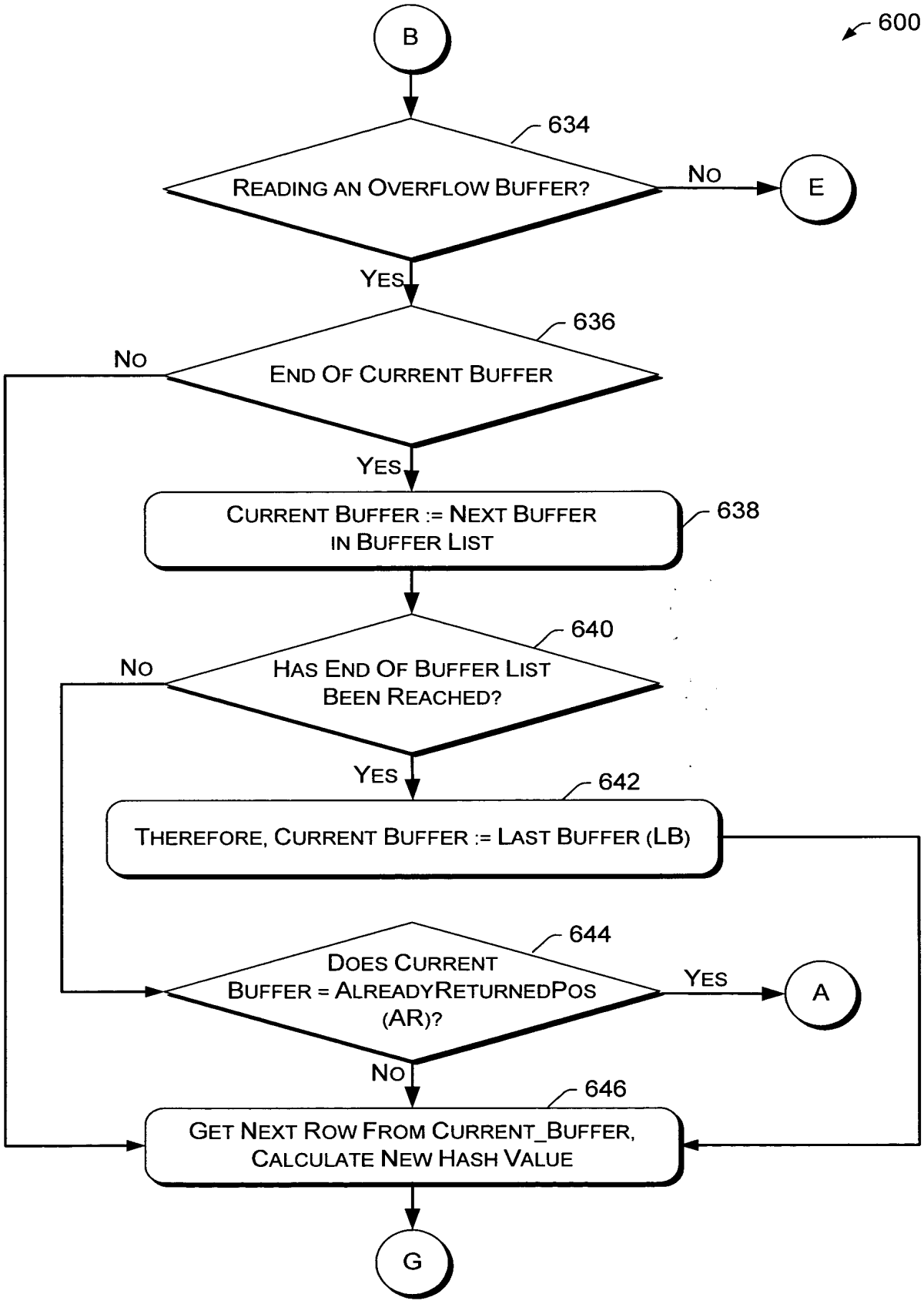
600
B
634
READING AN OVERFLOW BUFFER?
NO
E
YES
636
NO
END OF CURRENT BUFFER
YES
CURRENT BUFFER := NEXT BUFFER IN BUFFER LIST
638
640
NO
HAS END OF BUFFER LIST BEEN REACHED?
YES
642
THEREFORE, CURRENT BUFFER := LAST BUFFER (LB)
644
DOES CURRENT BUFFER = ALREADYRETURNEDPOS (AR)?
YES
A
NO
646
GET NEXT ROW FROM CURRENT_BUFFER, CALCULATE NEW HASH VALUE
G

Fig. 6c

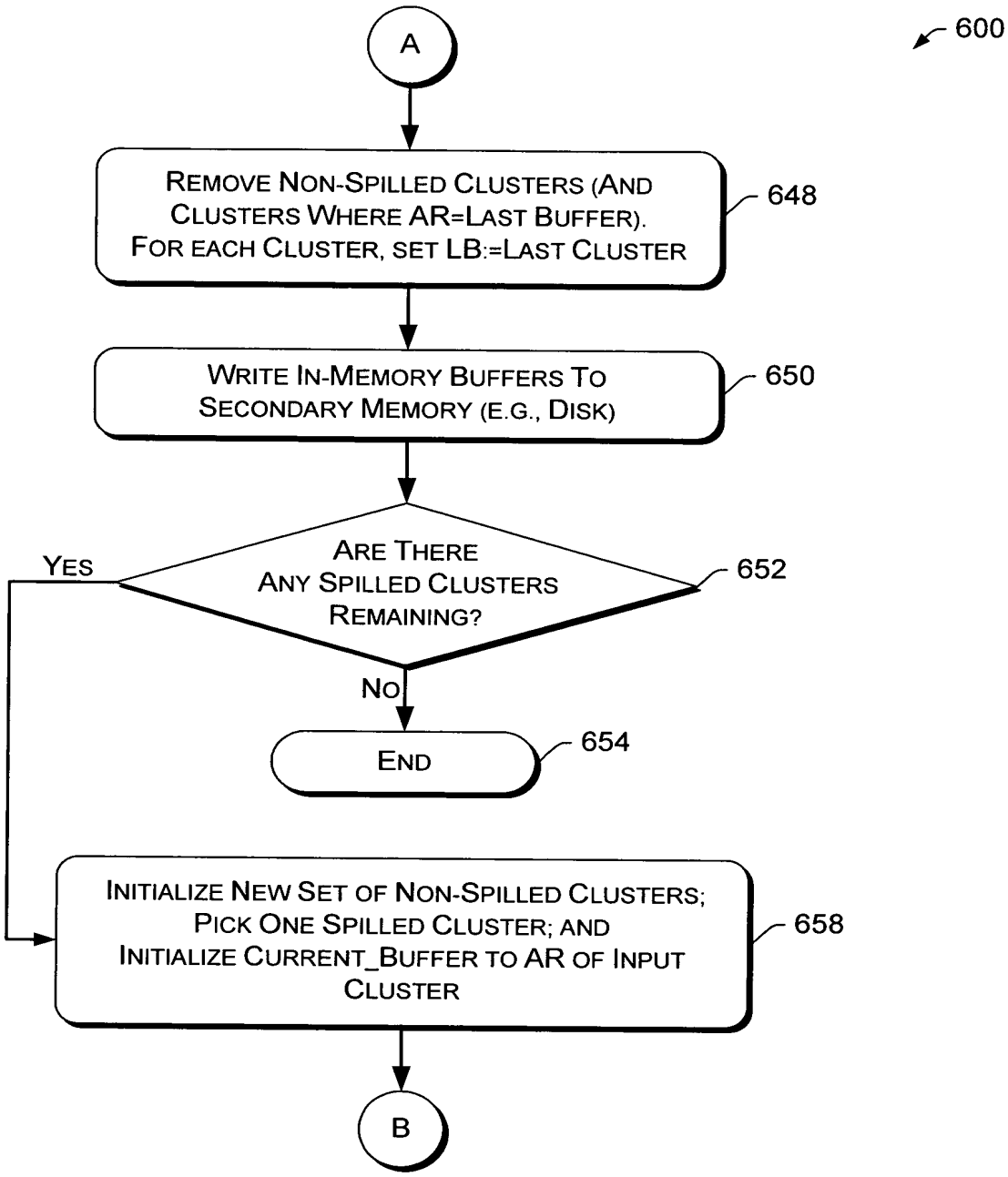
600
A
REMOVE NON-SPILLED CLUSTERS (AND CLUSTERS WHERE AR=LAST BUFFER). FOR EACH CLUSTER, SET LB:=LAST CLUSTER
648
WRITE IN-MEMORY BUFFERS TO SECONDARY MEMORY (E.G., DISK)
650
ARE THERE ANY SPILLED CLUSTERS REMAINING?
652
YES
NO
END
654
INITIALIZE NEW SET OF NON-SPILLED CLUSTERS; PICK ONE SPILLED CLUSTER; AND INITIALIZE CURRENT_BUFFER TO AR OF INPUT CLUSTER
658
B

Fig. 6d